pA 01-308

reference 2

(19)日本国特許庁（ＪＰ）　(12)**公表特許公報（Ａ）**　(11)特許出願公表番号

特表平11－514116

(43)公表日　平成11年(1999)11月30日

| (51)Int.Cl.[6] | 識別記号 | Ｆ Ｉ | |
|---|---|---|---|
| Ｇ０６Ｆ　11/14 | ３１０ | Ｇ０６Ｆ　11/14 | ３１０Ｂ |

審査請求　有　　予備審査請求　有　（全 40 頁）

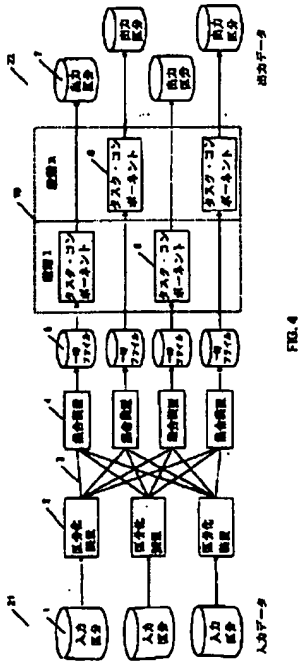
(21)出願番号　特願平9－523695
(86)(22)出願日　平成８年(1996)12月11日
(85)翻訳文提出日　平成10年(1998)６月11日
(86)国際出願番号　ＰＣＴ／ＵＳ９６／１９７０６
(87)国際公開番号　ＷＯ９７／２３８２６
(87)国際公開日　平成９年(1997)７月３日
(31)優先権主張番号　０８／５７０，５８５
(32)優先日　1995年12月11日
(33)優先権主張国　米国（ＵＳ）

(71)出願人　エーピー　イニティオ　ソフトウェア　コーポレーション
アメリカ合衆国　01742　マサチューセッツ州，コンコード，バージニア　ロード　477
(72)発明者　スタンフィル，クレッグ
アメリカ合衆国　02154　マサチューセッツ州，ワルサム，フローレンス　ロード　62
(74)代理人　弁理士　平木　祐輔　（外２名）

最終頁に続く

(54)【発明の名称】　コンポーネント・ベースの並列アプリケーションでチェックポイントを増加するための過剰区分化（ｏｖｅｒｐａｒｔｉｔｉｏｎｉｎｇ）システムおよび方法

(57)【要約】
コンピュータ・プログラムによって実行される作業を、チェックポイントがより頻繁に行われるようにさらに小さな片に区分化するための２つの方法である。当初、並列タスクはｑ個の初期区分を持つ１つまたは複数のデータ・セットで開始し、入力データ・セットを、区分化要素のなんらかの組合わせ(区分化装置／集合装置)によってｐ個の区分に分割し、データのｐ個の区分のそれぞれの上でコンポーネント・プログラムのインスタンスを実行し、１つまたは複数の出力ファイルのセットを作成し、各セットが区分化されたデータ・セットと見なされる。本発明は、新規の「過剰区分化」されたタスクを以下のように作成するためにこのようなタスクに適用される。（１）区分化装置は、整数過剰区分化係数ｎに対して、そのｑ個の入力をｎ＊ｐ個の区分に分割する「過剰区分化装置」と置換される。（２）コンポーネント・プログラムは、直列のｎ個の実行段階で実行され、コンポーネント・プログラムのｐ個のインスタンスは任意の時点で実行される。各段階では、コンポーネント・プログラムの各インスタンスが、入力データの１つの過剰区分



BEST AVAILABLE COPY

【特許請求の範囲】

１．（ａ）なんらかの整数過剰区分化係数ｎに対して、ｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割するステップと、

（ｂ）コンポーネント・プログラムの各インスタンスがｎ＊ｐ個のデータ区分の内の１つを読み取り、出力データの対応する区分を作成する、直列のｎ個の実行段階でコンポーネント・プログラムのｐ個のインスタンスを実行するステップと、

（ｃ）ｎ個の実行段階のそれぞれの最後でチェックポイント設定を可能にするステップと、

を含む、並列処理システムで実行可能なコンピュータ・アプリケーションでチェックポイントの頻度を増加するための方法。

２．ｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割するステップが、さらに、

（ａ）データのｎ＊ｐ＊ｑ個のチャネルを出力するように構成される区分化プログラムを、ｑ個のデータ入力に適用するステップと、

（ｂ）ｎ＊ｐ個のデータ区分を生成するために、区分化プログラムによって出力されるデータのｎ＊ｐ＊ｑ個のチャネルに集合プログラムを適用するステップと、

を含む、請求項１記載の方法。。

３．ｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割するステップが、さらに、

（ａ）ｑ個のデータ入力をｎ個のデータ・セグメントに分割するためにｐ個のカーソルを使用するステップと、

（ｂ）各データ・セグメントの最後を示すためにファイル終了信号を回帰的に生成するステップと、

を含む、請求項１記載の方法。。

４．並列処理システム上で実行可能なコンピュータ・アプリケーションでのチェックポイントの頻度を増加するためのコンピュータ・プログラムであって、

（ａ）なんらかの整数過剰区分化係数ｎに対して、ｑ個のデータ入力をｎ＊ｐ

個のデータ区分に機能的に分割する機能と、

（ｂ）コンポーネント・プログラムの各インスタンスがｎ＊ｐ個のデータ区分の内の１つを読み取り、出力データの対応する区分を作成する、直列のｎ個の実行段階でコンポーネント・プログラムのｐ個のインスタンスを実行する機能と、

（ｃ）ｎ個の実行段階のそれぞれの最後でチェックポイント設定を可能にする機能と、

を実行するためにコンピュータ・システムによって読み取られ、実行されると、コンピュータ・システムを構成する、コンピュータ・システムによって読取可能な媒体に記録されるコンピュータ・プログラム。

５．ｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割する機能が、さらに、

（ａ）データのｎ＊ｐ＊ｑ個のチャネルを出力するように構成される区分化プログラムを、ｑ個のデータ入力に適用する機能と、

（ｂ）ｎ＊ｐ個のデータ区分を生成するために、区分化プログラムによって出力されるデータのｎ＊ｐ＊ｑ個のチャネルに集合プログラムを適用するステップと、

を含む、請求項４記載のコンピュータ・プログラム。

６．ｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割する機能が、さらに、

（ａ）ｑ個のデータをｎ個のデータ・セグメントに分割するためにｐ個のカーソルを使用する機能と、

（ｂ）各データ・セグメントの最後を示すためにファイル終了信号を回帰的に生成する機能と、

を含む、請求項４記載のコンピュータ・プログラム。

７．（ａ）なんらかの整数過剰区分化係数ｎに対して、ｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割する機能と、

（ｂ）コンポーネント・プログラムの各インスタンスがｎ＊ｐ個のデータ区分の内の１つを読み取って、出力データの対応する区分を作成する、直列のｎ個の実行段階でコンポーネント・プログラムのｐ個のインスタンスを実行する機能と

（ｃ）ｎ個の実行段階のそれぞれの最後でチェックポイント設定を可能にする機能と、

を実行するために、記録媒体が特定の事前に定義された方法でコンピュータを動作させるように、並列処理システム上で実行可能なコンピュータ・アプリケーションでのチェックポイントの頻度を増加するためにコンピュータ・プログラムで構成されるコンピュータ読取可能な記録媒体。

８．ｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割する機能が、さらに、

（ａ）データのｎ＊ｐ＊ｑ個のチャネルを出力するように構成される区分化プログラムを、ｑ個のデータ入力に適用する機能と、

（ｂ）ｎ＊ｐ個のデータ区分を生成するために、区分化プログラムによって出力されるデータのｎ＊ｐ＊ｑ個のチャネルに集合プログラムを適用する機能と、

を含む、請求項７記載のコンピュータ読取可能な記録媒体。

９．ｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割する機能が、さらに、

（ａ）ｑ個のデータ入力をｎ個のデータ・セグメントに分割するためにｐ個のカーソルを使用する機能と、

（ｂ）各データ・セグメントを示すためにファイル終了信号を回帰的に生成する機能と、

を含む、請求項７記載のコンピュータ読取可能な記録媒体。

【発明の詳細な説明】

コンポーネント・ベースの並列アプリケーションでチェックポイントを増加するための過剰区分化（overpartitioning）システムおよび方法

発明の背景

技術分野

本発明は、コンピュータ処理システムに関し、さらに特定するとコンピュータ・プログラムによって実行される作業を、チェックポイントがさらに頻繁に実行されるように、さらに小さな片に区分化するための方法に関する。

関連技術の説明

単一プロセッサ・コンピュータの計算速度は、過去３０年間に渡って飛躍的に進歩してきた。しかしながら、多くの分野は、最高速度の単一プロセッサ・コンピュータさえも上回る計算速度を必要とする。例は、複数ユーザがコンピュータ・リソースに並行してアクセスし、応答時間はシステムが商業上受け容れられるためには遅くてはならないトランザクション処理である。別の例は、数百ギガバイトの情報が処理されなければならず、直列コンピュータ上でのデータ処理に数日または数週間を要する可能性のあるデータベース調査（mining）である。したがって、このような問題を取り扱うために多岐に渡る「並列処理」システムが開発されてきた。この説明のために、並列処理システムは、ローカル（例えば、ＳＭＰコンピュータのようなマルチプロセッサ・システム）つまりローカルに分散された（例えば、クラスタまたはＭＰＰとして結合される複数プロセッサ）、または遠隔に、つまり遠隔に分散された（例えば、ＬＡＮまたはＷＡＮネットワークを介して結合される複数のプロセッサ）、またはその任意の組み合わせで複数の中央演算処理装置（ＣＰＵ）を使用するコンピュータ・システムの任意の構成を含む。

さらに多くのビジネス・アプリケーションは、データをある種の様式で変換する

る「コンポーネント・プログラム」のセットを含む。例えば、賃金台帳アプリケーションには、個人別就労時間集計用紙を支払小切手に変換したプログラムが含まれる可能性がある。このような構成要素を並列処理システム上での実行のため

のアプリケーションの一部として使用するためには、以下が共通した慣行である。

（１）並行の度合いｐが選択される。大部分の場合、ｐはプロセッサの数と同一である。

（２）コンポーネント・プログラムに対する１つまたは複数の入力は、ｐ個の入力ファイルのセットの間で分割される（「区分」）。

（３）コンポーネント・プログラムのｐ個のインスタンスは、さまざまなプロセッサで同時に実行される。

（４）コンポーネント・プログラムの各インスタンスは、入力データの１つの区分を処理する。

コンポーネント・プログラムが実行を終了すると、その出力のそれぞれは、ｐ個の区分に分割される。その後、出力は、さらなるコンポーネント・プログラムのために入力として使用されるか、あるいは記憶装置に書き込まれる。

これらの区分内でのデータの編成は、大部分の場合、かなり重要である。つまり、多くの場合、データが適切に区分化されていないと、コンポーネント・プログラムは間違った結果を出す。さらに、各区分内のデータの順序がかなり重要であるのも、大部分の場合に当てはまる。例えば、多くのプログラムは、その入力（複数の場合がある）が何らかの様式でソートされることを必要とする。

データの正しい区分化を達成するためには、入力ファイルを取り、それをなんらかの規則に則ってｐ個の部分に分割する別個の「区分化装置（partitioner）」プログラムを使用するのが一般的な慣行である。区分化の方法がコンポーネント・プログラムによって出される結果に影響を及ぼさない場合では、区分化装置はデータを任意の様式で分割することができる。区分化が、各レコードをレコード内のデータ値に基づいてｐ個の区分の内の１つに割り当てる数学的な関数に基づいているのも一般的である。

一般的なケースでは、入力のそれぞれは、必ずしもコンポーネント・プログラムのニーズに一致した方法ではないが、ｑ個の初期区分に分割される。このような

なデータを区分化するためには、一般的な慣行とは、区分化装置プログラムのｑ

個のコピーを実行することであり、各コピーはｑ個の初期区分の１つを入力として与えられる。区分化プログラムのｑ個のコピーのそれぞれはｐ個の出力区分を作成するので、これは入力データをｐ＊ｑ区分に分割する。この問題を第２の「集合装置（gatherer）」プログラムを利用して、ｑ個の区分（区分化装置の各インスタンスに１つ）のグループを結合し、最終的な出力を出すことによって解決するのが一般的な慣行である。

アプリケーションは、いったん書き込まれると、任意の並列の度合いｐで実行可能となるのが望ましい。これは、時間が経過するに従って、さらに多数のプロセッサを備えるコンピュータの取得を必要とするデータの量が増加するために必要である。これを支持して、区分化装置／集合装置プログラムを、それらが任意の希望の数の区分ｐを作成できるように、およびコンポーネント・プログラムが正しく動作するように、パラメータ化することが一般的な慣行である。この特性が「スケーラビリティ」と呼ばれる。

図１は、ｑ個の初期区分をｐ個の出力区分に分割するための従来の技術の区分化装置／集合装置の図である。図１は、単一入力から読み取り、単一出力に書き込むコンポーネント・プログラムの場合のこのようなシステムを示す。入力データ２１は、ｑ＝３初期区分１に区分化されている。希望の並列の度合いはｐ＝２である。区分化装置プログラム２の３つのインスタンスが実行され、通信チャネル３に沿って伝送される６つの中間結果を出す。これらの中間結果は、集合装置プログラム４の２つのインスタンスによって読み取られ、２つの一時ファイル５を作成する。それから、コンポーネント・プログラム６の２つのインスタンスが実行され、その内のそれぞれが別個の一時ファイル５から読み取り、別個の出力ファイル７に書き込む。出力ファイル７は出力データ・セット２２の区分を構成する。もし複数の出力データ・セット２１があれば、区分化装置２、通信チャネル３、集合装置５、および一時ファイル６は入力データ・セットごとに一度複製される（区分化装置２の数を変えながら）。複数の出力データ・セット２２を持つことも可能である。最後に、いくつかの個別入力ファイルが区分化されるのではなく、コンポーネント・プログラム（図示されていない）のすべてのインスタ

ンスに一斉送信されることが考えられる。

このアーキテクチャには多数の変形がある。例えば、一時ファイル５が対応する通信チャネルによって置き換えられるか、あるいは通信チャネル３が対応するファイルで置き換えられる可能性がある。

図２は、「事前区分化済み(pre-partitioned)データ」の重要な特別ケースを示す従来の技術のシステムの図である。このケースは、入力区分数ｑがコンポーネント・プログラムの希望の並列度合いｐに等しく、入力データ２１が区分化される方法がコンポーネント・プログラム６に適当である場合に発生する。このケースでは、データを区分化する必要はなく、区分化装置２、通信チャネル３、集合装置４、および一時ファイル５を排除することができる。区分化装置／集合装置要素２－５の排除により、かなりの計算上の作業とかなりの記憶スペースが節約されるため、この特別ケースを利用することは有利である。

コンポーネント・プログラム６が複数の入力を持つ場合、その入力の内のいくつかは、図１に図示される区分化を必要とし、それ以外は、図２でのように「事前区分化」されることが考えられる。

複数のコンポーネント・プログラムを備える複雑なアプリケーションを構築するために、人は、図１または図２に示されるような構造を持つ複数の計算単位（「タスク」）をリンクさせる。このリンクは、処理のある段階の出力区分７を、次の段階の対応する入力区分１に割り当てることによって実行される。

複雑なアプリケーションの場合は、システム故障が発生しても、アプリケーション全体を始めから実行し直すのではなく、アプリケーションが計算の中間状態を捕捉する「チェックポイント」から続行できるようにする「チェックポイント」機構を備えることが望ましい。チェックポイントに関する一般的な説明、および本発明が貴重となる可能性がある改善型チェックポイント・システムの具体的な例については、本発明の譲受け人に譲渡された「計算の状態を再構築するための方法およびシステム（Methods and Systems for Reconstructing the State of a Computation）」と題する同時係属の特許明細書を参照する。

チェックポイントの制約の内の１つとは、システムが「静止」している場合、つまりアプリケーション・プログラムが実行していない場合にだけチェックポイ

ントが作成されるという点である。システムは、タスクの間で静止しており、したがって各タスクの端でチェックポイントを作成できるように配置できる。また、集合装置プログラム４が実行を終了した直後に、タスクにとって内部に１つのチェックポイントを作成することも可能である。ただし、これらの技法がチェックポイントに十分に頻繁な機会を生み出さない可能性もあると考えられる。例えば、タスクのコンポーネント・プログラム６が１０時間実行すると、アプリケーションは、どのチェックポイントも可能ではないときに必然的に１０時間という期間を持つだろう。その１０時間の間の任意の時点で故障が発生すると、アプリケーション全体を最初から起動し直さなければならない。

したがって、並列構成要素ベースのアプリケーション内でさらに多くの頻繁なチェックポイントを許す方法に対するニーズがある。本発明は、並列処理システムで実行中のアプリケーションに特に有効であり、分散処理システム上で実行するアプリケーションにも有効であるこのような方法を提供する。

発明の要約

本発明は、複雑な並列アプリケーション内で、それがさらに頻繁に静止（したがって、チェックポイント設定可能（checkpointable））となるように、タスクを修正するための２通りの方法を含む。

当初、並列タスクはｑ個の初期区分を持つ１つまたは複数の入力データ・セットで開始し、入力データ・セットを、区分化要素（つまり、区分化装置／集合装置）のなんらかの組み合わせによってｐ個の区分に分割し、データのｐ個の区分のそれぞれでコンポーネント・プログラムのインスタンスを実行し、１つまたは複数の出力ファイルを作成し、そのそれぞれのセットが区分化されたデータ・セットと見なされる。本発明は、新規の「過剰区分化」されたタスクを以下のように作成するためにこのようなタスクに適用される。

（１）区分化装置は、整数過剰区分化係数ｎのためにそのｑ個の入力をｎ＊ｐ個の区分に分割する「過剰区分化装置」と置換される。

（２）コンポーネント・プログラムは、ｎ個の実行段階が直列に実行され、コンポーネント・プログラムのｐ個のインスタンスは任意の時点で実行される。各段

階では、コンポーネント・プログラムの各インスタンスが、入力データの１つの過剰区分(overpartition)を読み取り、出力データの１つの区分を作成する。

（３）ｎ個の実行段階のそれぞれの最後で、システムは静止しており、チェックポイント設定をすることができる。

本発明は２つの好ましい実施例を含む。１つの実施例とは、過剰区分化された中間ファイルを作り出すために、既知の区分化装置プログラム、通信チャネル、および集合装置プログラムを使用して入力データを「明示的に過剰区分化」することである。この実施例は、図１に図示される形式のタスクに適用可能である。第２実施例は、オリジナル入力データの連続サブセットを連続的に読み取るためにコンポーネント・プログラムを手配することによって入力データを「動的に過剰区分化する」ことである。この実施例は、図２に図示される形式の多くのタスクに適用可能である。

本発明の好ましい実施例の詳細は、添付図面および以下の説明に述べられている。いったん本発明の詳細が既知になると、数多くの補助的なイノベーションおよび変化が当業者に明らかになるだろう。

図面の簡単な説明

図１は、ｑ個の初期区分をｐ個の出力区分に分割するための従来の技術による区分化装置／集合装置の図である。

図２は、「事前区分化データ」の特別ケースを示す従来の技術のシステムの図である。

図３ａは、本発明に従った過剰区分化システムの適用の前のオリジナル・タスクの全体的な図である。

図３ｂは、本発明に従った過剰区分化システムの適用後の図３ａのタスクの全体的な図である。

図４は、本発明に従った明示過剰区分化システムの好ましい実施例を示す、図３ｂの過剰区分化システムのより詳細な図である。

図５ａおよび図５ｂは、データ・フロー、および本発明に従った明示過剰区分化システムを実現するためのシステムのプロセッサ、通信、および記憶域の関係

性を示すさらに詳細な図である。

図６は、本発明に従った動的過剰区分化システムの好ましい実施例を示す、図３ｂの過剰区分化システムのさらに詳細な図である。

図７は、データ・フロー、および本発明に従った動的過剰区分化システムを実現するためのシステムのプロセッサ、通信および記憶域の関係性を示す、さらに詳細な図である。

図８は、本発明に従った動的過剰区分化システムでカーソルおよびモニタを実現するためのシステムのデータ・フロー、機能、および通信の関係性を示す図である。

さまざまな図の類似した番号および名称は、類似した要素を示す。

本発明の詳細な説明

本説明を通して、好ましい実施例および図示される例は、本発明に対する制限としてより、むしろ例と見なされる必要がある。

概要

図３ａは、本発明に従った過剰区分化システムの適用の前のオリジナル・タスクの全体的な図である。図３ａは、ｑ個の入力区分１を備える１つまたは複数の入力データ・セット２１で開始し、それらを区分化要素８のなんらかの組み合わせ（つまり、区分化装置／集合装置）によってｐ個の区分１２に分割し、データのｐ個の区分１２のそれぞれでコンポーネント・プログラム６のインスタンスを実行し、出力データ２２の１つまたは複数の出力区分７を作成する並列タスク１１を示す。

図３ｂは、本発明に従った過剰区分化システムの適用後の、図３ａのタスクの全体的な図である。新規の過剰区分化されたタスク１３は、以下のように作成される。

（１）区分化装置８は、なんらかの整数過剰区分化係数ｎのために、そのｑ個の入力区分をデータのｎ＊ｐ個の区分１２に分割する「過剰区分化装置」９と置換される。

（２）コンポーネント・プログラム６は、ｎ個の実行段階１０が直列に実行され

、コンポーネント・プログラムのｐ個のインスタンスは任意の時点で実行される。各段階で、コンポーネント・プログラム６の各インスタンスは、入力データ２１の１つの区分１２を読み取り、出力データ２２の１つの区分７を作成する。入力データ・ファイル２１は、順次、階層、関係などの好ましいフォーマットとすることができる。

（３）ｎ個の実行段階のそれぞれの最後では、システムは静止しており、チェックポイント設定をすることができる。

ここでの重要な特徴とは、大部分の場合には、コンポーネント・プログラム６を実行するために必要とされる時間が処理されるデータの量に強く結び付けられている、例えば、データの半分は約半分の時間で処理することができるという点である。したがって、データの量のｎ倍の削減は、対応する係数ｎ分、コンポーネント・プログラム６のインスタンスの実行時間を縮小し、結果的に係数ｎ分、チェックポイント設定機会の間の間隔を削減する可能性がある。

本発明は、２つの好ましい実施例を含む。１つの実施例は、過剰区分化された中間ファイル５を作成するために、既知の区分化プログラム２、通信チャネル３、および集合装置プログラム４を使用して入力データ２１を「明示的に過剰区分化」することである。この実施例は、図１に図示される形式のタスクに適用可能である。第２実施例は、オリジナルの入力データ２１の連続サブセットを連続的に読み取るために、コンポーネント・プログラム６を手配することによって入力データ２１を「動的に過剰区分化する」ことである。この実施例は、図２に図示される形式の多くのタスクに適用可能である。

明示過剰区分化

明示過剰区分化の方法は、中間「過剰区分化された」ファイル５のセットを作り出すために既存の区分化装置２、通信チャネル３、および集合装置４を使用してから、過剰区分化されたファイルと対照して複数の実行段階でコンポーネント・プログラム６を実行する。この方法は、以下の要件が満たされる場合に使用される。

（１）方法は、入力データ・セット２１、並列の希望の度合いｐ、および過剰区

分化の希望の度合いｎからｑ個の入力区分１を提示されなければならない。

（２）区分化装置プログラムｐが、少なくともｎ＊ｐ個の出力区分を作り出すために有効に構成されることが必要である。

（３）コンポーネント・プログラム６が「スケーラブル」であることが必要である。つまり、アプリケーションが並列の任意の度合いｐに対して有効な結果を出すことが理解されなければならない。

図４は、図３ｂの過剰区分化システムのさらに詳細な図であり、本発明に従った明示過剰区分化システムの好ましい実施例である。前記事実が真実の場合、明示過剰区分化は、以下のように実現できる。

（１）区分化装置２のｑ個のインスタンスが、ｑ個の入力区分１と作用するように設定される。区分化装置２の各インスタンスは、ｎ＊ｐ個の区分を作成するために構成される。

（２）集合装置４のｎ＊ｐ個のインスタンスは、区分化装置２の出力と作用するように設定される。

（３）区分化装置２のｑ個のインスタンスは、図示されるように、ｎ＊ｐ＊ｑ個の通信チャネル３を使用してｎ＊ｐ個のインスタンスと接続される。

（４）すべての区分化装置２および集合装置４が実行され、ｎ＊ｐ個の一時ファイル５を作成する。

（５）ｎ個の実行段階１０は、前記のように実行される。各段階では、コンポーネント・プログラム６のｐ個のインスタンスが実行され、それぞれのこのようなインスタンスは、従来の技術におけるように、一時ファイル５の内の１つからデータを消費し、データを出力区分ファイル７の内の１つに書き込む。出力は、ｎ＊ｐ個のファイルを構成する。

多くのケースでは、構成要素プログラム６の論理は、ｎ個の連続実行段階１０によって作り出される出力が、連結される場合には、ｎの値に関係なく、非過剰区分化されたケースでのように（ｎ＊ｐよりむしろ）ｐ個の出力ファイルの同じセットを作り出すようなものある。このケースでは、ｐ個の出力ファイルのセットは、コンポーネント・プログラム６の各インスタンスによってその連続実行段

階１０の間に出される出力を追加するだけで作成することができる。これには、管理されなければならない出力ファイルの数を削減するという利点がある。

従来の技術でのように、このアーキテクチャには数多くの変形がある。例えば、一時ファイル５が対応する通信チャネルによって置換されたり、通信チャネル３が対応するファイルと置換することができる。

前記ステップは、手動によって（例えば、アプリケーションの全体的な実行を制御するために使用されるソフトウェアを書き直すことによって）または自動的に実現できる。好ましい実施例では、自動ケースは、タスクごとに以下の情報（または同等な情報）がコンピュータで読取り可能な形式で指定されることを必要とする。

（１）並列の希望する度合いｐ、および過剰区分化の希望する度合いｎ

（２）コンポーネント・プログラム６のアイデンティティ

（３）コンポーネント・プログラム６への入力データ２１を有するファイルのアイデンティティ

（４）タスクの入力データ・セット２１ごとに、以下の追加情報：

１）その入力区分１を有するファイルのアイデンティティ

２）その入力に適切な区分化装置２のアイデンティティ

３）その入力に適切な集合装置４のアイデンティティ

４）ｐ個の一時ファイル５を記憶することができる適当な記憶装置のアイデンティティ

（５）出力データ・セット２２ごとに、ｐ個の出力区分ファイル７を記憶することができる適当な記憶装置のアイデンティティ

（６）区分化装置２を実行することができるｑ個の「区分」プロセッサの識別子

（７）集合装置４を実行することができるｐ個の「集合」プロセッサの識別子（これらは、希望する場合、区分プロセッサと同じとすることができる）。

（８）コンポーネント・プログラム６を実行することができるｐ個の「ワーク」プロセッサの識別子（これらは、希望する場合、区分プロセッサまたは集合プロセッサと同じとすることができる）。

（９）区分化装置２、集合装置４、およびコンポーネント・プログラム６の場合

必要に応じてそれらを実行させ、それらをデータ・ファイルおよび通信チャネルとして接続させるためのこのような情報、および区分化装置２のケースでは、すべて既知の様式で任意の数の区分を作り出すためにそれらをどのようにして構成するのかに関する情報。好ましい実施例では、この情報は、接続先のデータ・ファイル／サブルーチンのアイデンティティ（加えて、区分化装置２の場合、区分の数）が指定され、希望すればプログラムを実行する「引数ライン・パラメータ」のシーケンスを戻すサブルーチンとして表される。

いったん上記情報（またはその均等物）が提供されると、明示過剰区分化アルゴリズムを実行することができる。図５ａおよび図５ｂは、データ・フロー、および本発明に従って明示過剰化区分化システムを実現するためのシステムのプロセッサ、通信、および記憶域の関係性を示す詳細な図である。

図５ａを参照すると、タスクのために入力データ２１を有するファイルが、好ましい実施例で次に示すように反復的に処理される。

（１）手元にあるデータ入力ファイル２１の入力区分１の数ｑを識別すること。

（２）ｎ＊ｐ＊ｑ個の通信チャネル３を下記によって作成すること、

１）第１区分プロセッサ２３で第１のｎ＊ｐ個のチャネルを起動し、次の区分プロセッサ２３などで次のｎ＊ｐ個のチャネルを起動すること、

２）サイクルが第１集合プロセッサ２４で開始し直して繰り返すｎ＊ｐ個の通信チャネルが割り当てられるまで、第１のｎ個のチャネル３を第１集合プロセッサ２４に接続し、次のｎ個のチャネルを次の集合プロセッサ２４などに接続すること、

（３）区分化装置２のｑ個のインスタンスを実行すること、ただし、

１）各インスタンスが区分プロセッサ２３の１つに割り当てられる。

２）各インスタンスが、ｎ＊ｐ個の出力区分を作成するために構成される。

３）各インスタンスが、そのプロセッサ内で起動する通信チャネル３のすべてに接続する。

４）ｉ番目のインスタンスが、入力データ・セット２１のｉ番目の入力区分１か

ら読み取る。

（４）集合装置４のｎ＊ｐ個のインスタンスを実行すること、ただし、

１）ｎ個のインスタンスが集合プロセッサ２４のそれぞれに割り当てられる。各プロセッサ上のｉ番目のインスタンスが、未接続の通信チャネル３がなくなるまで、ｉ番目の入信通信チャネル３、ｎ＋ｉ番目のチャネル３、２ｎ＋ｉ番目のチャネル３などに接続する。

２）ｋ番目の集合プロセッサ２４上で実行するインスタンスが、「一時ファイル」記憶装置２５のセット内の記憶装置にｋ番目の一時ファイル５を書き込む（さらに高い性能のために、ｋ番目の集合プロセッサ２４で実行する各インスタンスは、別個のｋ番目の記憶装置上のｋ番目の一時ファイル５に書き込む）。

（５）実行を終了するために区分化装置２と集合装置４のすべてのインスタンスを待機してから、通信チャネル３を削除すること、

（６）入力データ・ファイル２１の次のセットに継続する前に、オプションでシステムにチェックポイントを設定すること（この時点で、手元のデータ入力ファイル２１の過剰区分化は完了している。チェックポイントの設定が、必ずしも、各チェックポイント設定可能なイベントで実際に実行される必要はないことに注意する）。

（７）入力データ・ファイル２１の任意の次のセットのために繰り返されるステップ（１）－（６）。

図５ｂを参照すると、ｎ個の実行段階１０は、好ましい実施例で次に示すように一時ファイル５内のデータに対して実行される。

（１）コンポーネント・プログラム６のｐ個のコピーを、ｎ個の実行段階１０のそれぞれの間にワーク・プロセッサ２６の内のそれぞれで１つ実行すること。

（２）ｉ番目の実行段階１０の間に、ｊ番目のワーク・プロセッサ２６上で実行される、コンポーネント・プログラム６のインスタンスが、一時記憶装置２５からｉ番目の一時ファイル５を読み取り、出力記憶装置２６上のｉ番目の出力区分ファイル７に書き込む（さらに高い性能のために、ｊ番目のワーク・プロセッサ２６上で実行する各インスタンスは、ｊ番目の一時記憶装置からｉ番目の一時フ

ファイル５を読み取り、別個のｊ番目の出力記憶装置２７上のｉ番目の出力区分ファイル７に書き込む）。

（３）オプションで、一時ファイル５をそれらが消費される実行段階１０内で削除すること。

（４）システムにチェックポイントが設定される、ｎ個の実行段階１０のそれぞれで実行するコンポーネント・プログラム６のすべてのｐ個のインスタンスの終了を待機すること。

このようにして、ｎ個の実行段階１０のそれぞれの最後で、コンポーネント・プログラム６のｐ個のインスタンスは、合計入力データ２１のｎ分の１を処理してから、静止し、チェックポイントを設定することができる。例えば、ｑ＝２およびｐ＝５の場合、従来の技術の下では、各コンポーネント・プログラム６は、１サイクル時間内で合計入力データ２１の５分の１を処理するだろう。つまり、すべてのコンポーネント・プログラム６の並行処理を想定し、データのすべてが同じサイクル時間内に処理される。ただし、チェックポイント設定は、サイクルが終了するまで発生しない。本発明を使用し、３という過剰区分化係数を選択することにより、処理対象の区分の総数は３＊５＝１５となる。ただし、それぞれが従来の技術の区分の約３分の１のサイズの５個の区分だけが各実行サイクルで処理される。線形処理時間を想定し、各コンポーネント・プログラム６のサイクル時間は、従来の技術のサイクル時間の約３分の１となる。チェックポイント設定が、このように縮小された各サイクルの後に実行されてから、次の実行段階１０が開始される。チェックポイント設定段階および実行段階１０は、それからすべての過剰区分が処理されるまで繰り返される。

動的過剰区分化

前記のように、いくつかのケースでは、入力データ２１は、コンポーネント・プログラム６に協和的な方法で初期に区分化され、そのケースでは図２に図示される最適化が可能になる（つまり、区分化装置２、通信チャネル３、集合装置４、および一時ファイル５が排除される）。明示過剰区分化方法は、必然的に、区分化装置などの使用を必要とするため、事前区分化された最適化の利点を失う。

本発明の第２実施例は、一時的な過剰区分化されたファイル５を作成するために区分化装置などに頼らない。代わりに、第２実施例は、効果的に適所の入力データ・ファイル１を区分化し直す。この方法が「動的過剰区分化」と呼ばれる。

図６は、図３ｂの過剰区分化システムのさらに詳細な図であり、本発明に従った動的過剰区分化システムの好ましい実施例を示す。この方法は、コンポーネント・プログラム６が受け容れることができる区分が、入力ファイル１をデータ・セグメントと呼ばれる順次区分２７に分割することによって入手される場合に使用される（このようなセグメントを作成する方法は、以下に説明される）。

動的過剰区分化動作２８は、コンポーネント・プログラム６が入力区分１のデータ・セグメント２７にアクセスできるようにするという効果を持ち、各入力区分１からの１つのセグメントが、明示過剰区分化に類似した方法で、複数の実行段階１０のそれぞれのコンポーネント・プログラム６によって処理される。動的過剰区分化動作２８は、一時ファイルを作成しないでこれを実行する。動的過剰区分化の本質とは、コンポーネント・プログラム６の各インスタンスを、入力データ・ファイル１のデータ・セグメント２７を読み取らせるが、このようなインスタンスがそれが完全なデータ・セット１を読み取っていると考えることができるように「騙す」ことである。

図７は、データ・フロー、および本発明に従った動的過剰区分化を実現するためのシステムのプロセッサ、通信、および記憶域の関係性を示すさらに詳細な図である。動的過剰区分化は、次に示すように、好ましい実施例において実現される。

（１）入力区分１ごとに固定「カーソル」２９およびモニタ３０を作成すること（図６の動的過剰区分化動作２８は、カーソル２９およびモニタ３０を有する。カーソル２９は、入力区分１内の位置（例えば、バイト・オフセット）を記憶する。これによって、コンポーネント・プログラム６は、カーソル２９によって現在指されている位置で入力ファイル１からデータを読み取ることができる。モニタ３０は、カーソル２９によって指されるデータ・レコードを検査し、適当なときにカーソル２９に、入力区分１内の次のデータ・レコードを返すのではなく「

ファイル終了」（ＥＯＦ）を示すように命令することができる。）
（２）各実行段階１０のコンポーネント・プログラム６のｐ個のインスタンスを実行すること。
（３）（コンポーネント・プログラム６が、入力区分１に直接にアクセスすることができるようにするのではなく）カーソル２９を介するだけでデータ・セグメント２７のデータにアクセスすること。
（４）モニタ３０がデータ・セグメント２７の終わりを検出すると、以下にさらに説明するようにＥＯＦを示すようにカーソル２９に指示すること（カーソル２９は実際には入力区分１の最後にはないため、このイベントは「人工的なファイル終了」と呼ばれる）。
（５）（そのそれぞれが、対応するデータ・セグメント２７内のデータのすべてを処理した）コンポーネント・プログラム６を終了すること。
（６）好ましくはチェックポイント設定システムの一部を構成する２段階コミット・プロトコルを使用して、それ以降、カーソル２９およびモニタ３０の内容を固定記憶域に書き込むこと。
（７）オプションで、次のデータ・セグメント２７に進む前にシステムにチェックポイントを設定すること。
（８）次の実行段階１０の開始時に、カーソル２９およびモニタ３０の内容をワーク記憶域に読み取ること。
（９）データにアクセスするために対応するカーソル２９を使用してコンポーネント・プログラム６のさらなる実行を開始すること（各カーソル２９は、ここでは入力区分１の中間部分、さらに具体的にはその入力区分１のデータ・セグメント２７の次のセグメントの開始を参照する。したがって、処理はまさにそれが前の実行段階１０で終了したところで起動する。各実行段階１０の出力は、各事前実行段階出力に追加され、休止なしに最初から最後まで入力データ・ファイル２１を通ってアプリケーションが進んだように見えるようにする。）
（１０）すべてのカーソル２９がその入力区分１の「本当の」最後に到達したときにすべての実行を終了する（つまり、すべてのデータが処理されればそれ以上

の段階が実行されない）。

モニタ３０がカーソル２９に人工的なＥＯＦを信号で知らせるように指示するときの決定は、コンポーネント・プログラム６によって利用される区分化機構に依存する。複数のケースがある。

最初に、コンポーネント・プログラム６の入力データが任意の様式で区分化されることが考えられる。これは、例えば、あるレコードの処理が完全に次のレコードの処理から独立している場合に発生することがある。このケースでは、セグメント境界は任意の便宜的なレコード境界に該当する。

第２に、コンポーネント・プログラム６は、なんらかのキー（例えば、従業員番号）の同じ値が設定されるすべてのレコードが同じセグメント２７に割り当てられることを要求する場合が考えられる。このケースでは、入力データ１は、そのキーに応じてソートされる必要がある。このケースでは、セグメント境界は、同一のキー値が設定される連続レコードの実行の間の境界に該当するように制約されるのが好ましい。

第３に、コンポーネント・プログラム６が複数の入力から読み取ることが考えられ、指定されたキー値が設定されるすべてのレコードが同じ実行段階で処理されることが必要とされる。このケースでは、すべてのこのような入力ファイル中のデータは、入力キーでソートされる必要があり、コンポーネント・プログラム６に先行するさまざまなモニタ３０は、全てのカーソル２９が、同じキー値境界上でそのデータを区分するように、人工的なＥＯＦ信号を調整しなければならない。

最後に、モニタ３０が人工的なＥＯＦを導入するために適当な場所を探し始めるときを判断するためにはなんらかの機構が必要とされる。動的過剰区分化の場合では、過剰区分化の度合いが事前に決定される必要がないことに注意する。コンポーネント・プログラム６は、いくつかのイベントが発生するまで実行される（例えば、特定のデータ値が入力データ・ストリーム内で発生するか、指定された時間間隔が期限切れになるか、指定された数のバイトまたはレコードが処理されるか、あるいは入力データ・ファイル・サイズの指定されたパーセンテージが

処理されるかなど）。

これらの要件は、本発明に従った動的過剰区分化システム内でカーソル２９およびモニタ３０を実現するためのシステムのデータ・フロー、機能、および通信の関係性を示す図である図８に図示されるアーキテクチャによって満たされる。

（１）すべてのモニタ３０は、相互通信のためにグローバルな「トリガ・チャネル」３１を共用する。

（２）特定のキー値で「調整」しなければならないすべてのモニタ３０は、調整チャネル３２を介して接続される（複数の調整チャネル３２がある場合がある）。

（３）任意のモニタ３０は、任意の時点で、トリガ・チャネル３１上で「位相破損」メッセージを一斉通信する。メッセージは、トリガ・チャネル３１上ですべての他のモニタ３０によって受け取られる。典型的には、このようなイベントは、ある一定期間の時間が経過した後に、一定のデータ量がモニタを通り抜けた後に、あるいは何らかのキー値がレコード内で見られた後に起こる。

（４）いったん位相破損メッセージが受信されると、各モニタ３０は、そのカレント・データ・セグメント２７から実際に読み取られた最後のレコードからキーを入手し、その調整チャネル３２上でそのキーを指定される「マスタ・モニタ」に伝送する。

（５）各マスタ・モニタは、１つのキーが（それ自体を含む）その調整チャネル３２上で各モニタ３０から受け取られるまで待機してから、このセットからの最大キー値を検出する。このキーが「ブレークポイント・キー」と呼ばれる。

（６）各マスタ・モニタは、その調整チャネル３２上ですべてのモニタ３０にブレークポイント・キーを一斉通信する。

（７）各モニタ３０は、それからその対応するカーソル２９が、ブレークポイント・キーの値を超えるキーに到達するまで対応するコンポーネント・プログラム６にデータを送達し続ける。このポイントでは、人工的なＥＯＦ信号がそのモニタ３０によって作成され、現在の実行段階１０の最後につながる。

（８）（コンポーネント・プログラム６が単一データ入力ファイル２１から読み

取る場合などの）モニタを調整する必要がない特別なケースでは、各調整チャネル３２が単一モニタ３０を含む（つまり、モニタ間での情報の共用は必要とされない）。上記アルゴリズムは、それ以外の場合変更されない。代替形式では、調整チャネル３２はまったく設定されない。

（９）（経過した時間、またはレコードの数などに基づくような）キーに関係なく位相破損が作成される特別なケースでは、モニタ３０は、トリガ・チャネル３１上で位相破損メッセージの受信直後に人工的なＥＯＦを作成するだけである。

カーソル２９およびモニタ３０の組み合わせは、少なくとも以下の３通りの内の１つで、コンポーネント・プログラム６と入力区分１の間におくことができる。

（１）コンポーネント・プログラム６は、カーソル２９およびモニタ３０を実現するサブルーチン・ライブラリを使用して書き込まれるか、あるいは再度書き込まれる。これには、コンポーネント・プログラム６用のソース・コードへのアクセスが必要とされ、時間を消費する可能性があるが、最高の性能を達成する可能性がある。

（２）コンポーネント・プログラム６は、既知の様式で基礎を成す入出力機構への呼出しに割り込む実行時ライブラリに対照してリンクされる。例えば、「開く」ための呼出しが割り込まれ、「カーソルを開く」ための呼出しに変換される。「読み取る」ための呼出しが妨害され、「カーソルから読み取る」ための呼出しに変換される。これには、コンポーネント・プログラム６のオブジェクト・コードへのアクセスが必要になるが、かなりの効率が見込まれる。

（３）「アダプタ・プロセス」が各コンポーネント・プログラム６とその入力区分１の間におかれる。アダプタ・プロセスは入力区分１ファイルをカーソル／モニタ機構を介して読み取ってから、データを通信チャネルまたは一時ファイルを介して既知の様式でコンポーネント・プログラム６に転送する。このオプションは、構成要素プログラム６の実行可能な（バイナリ）形式へのアクセスを必要とするだけであるが、入力区分１ファイルから通信チャネルまたは一時ファイルにデータをコピーする費用のためにやや非効率的となる可能性がある。

要約すると、コンポーネント・プログラム６の各インスタンスを、一時に入力データ・ファイル１の１つのデータ・セグメント２７だけを読み取るように「騙し」てから、各データ・セグメント２７の最後で人工的なＥＯＦを作成することによってコンポーネント・プログラム６の「自然」終了を作成することにより、コンポーネント・プログラム６の終了を強制することによって、アプリケーションはさらに頻繁に静止することになり、このようにして頻繁なチェックポイント設定を本質的にサポートしないアプリケーションのさらに優れたチェックポイント設定を可能にする。

本発明は、このようにしてそれがさらに頻繁に静止（したがってチェックポイント設定可能）するように複雑な並列アプリケーションでタスクを修正するための２通りの方法を含む。本発明は、並列処理システム上で実行するアプリケーションに特に有効であり、分散処理システムで実行しているアプリケーションにも有効であるこのような方法を提供する。

本発明は、ハードウェアまたはソフトウェア、あるいは両方の組み合わせで実現することができる。ただし、好ましくは、本発明は、それぞれがプロセッサ、（揮発性メモリと不揮発性メモリ、または記憶装置要素、あるいはそのすべてを含む）データ記憶装置システム、少なくとも１つの入力装置、および少なくとも１つの出力装置を含む、プログラマブル・コンピュータ上で実行するコンピュータ・プログラムで実現される。プログラム・コードは、本明細書中に記述される機能を実行し、出力情報を生成するための入力データに適用される。出力情報は、既知の様式で１つまたは複数の出力装置に適用される。

各プログラムは、コンピュータ・システムと通信するために高水準手順プログラミング言語またはオブジェクト指向型プログラミング言語で実現されるのが好ましい。ただし、プログラムは、希望する場合、アセンブリ言語または機械語で実現することができる。いずれにせよ、言語はコンパイルされた言語または翻訳された言語とすることができる。

このような各コンピュータ・プログラムは、記録媒体または装置が、本明細書中に説明される手順を実行するためにコンピュータによって読み取られるときに

、コンピュータを構成し、動作するための汎用または特殊目的プログラマブル・コンピュータによって読取可能な（ＲＯＭまたは磁気ディスケットのような）記録媒体または記録装置上に記録されるのが好ましい。発明的なシステムは、そのように構成される記録媒体によってコンピュータが本明細書中に説明される機能を特殊な事前に定義された方法で動作させる、コンピュータ・プログラムで構成されるコンピュータ読取可能な記録媒体として実現されるとも考えられる。

本発明の多くの実施例が説明された。それにも関わらず、本発明の精神および適用範囲から逸脱することなく、さまざまな修正を加えることができることが理解されるだろう。したがって、本発明が特に説明された実施例によって制限されるのではなく、添付された請求の範囲によってだけ制限されることが理解されなければならない。

【図1】

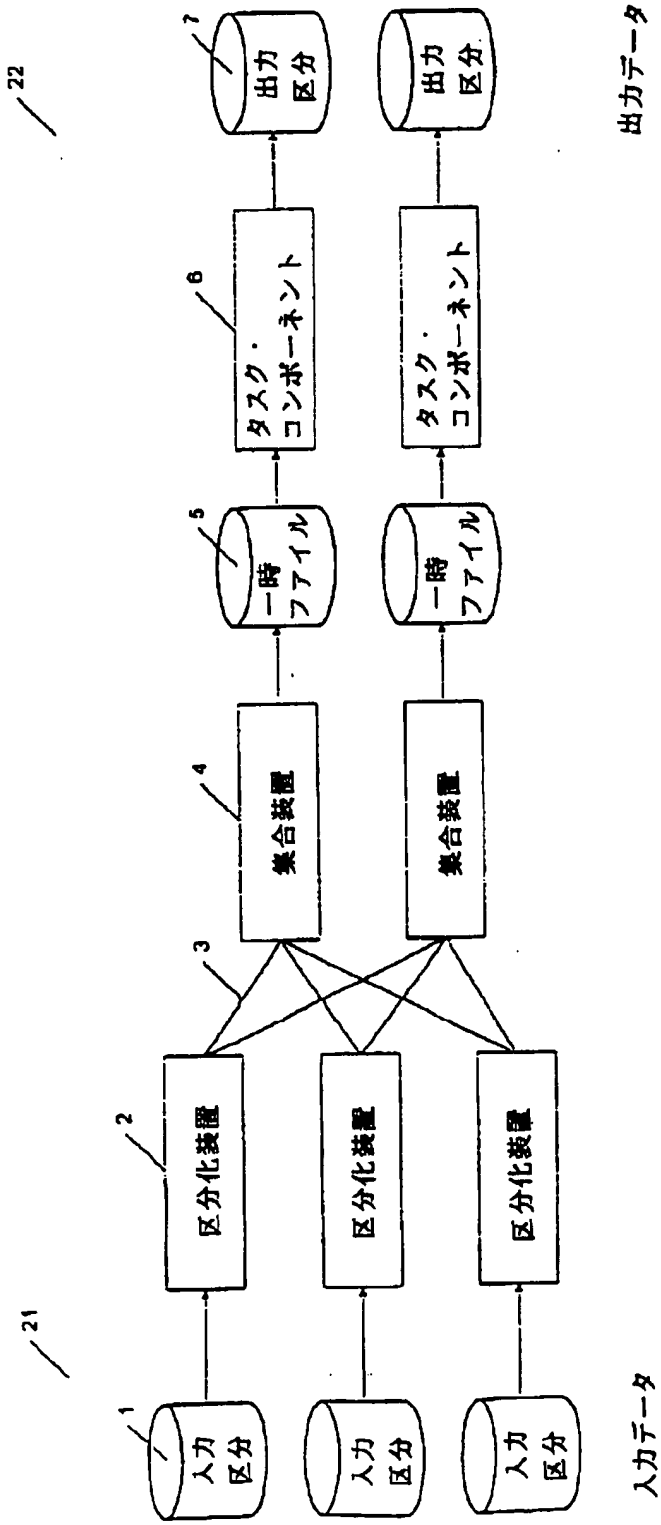



FIG. 1
(従来の技術)

【図2】

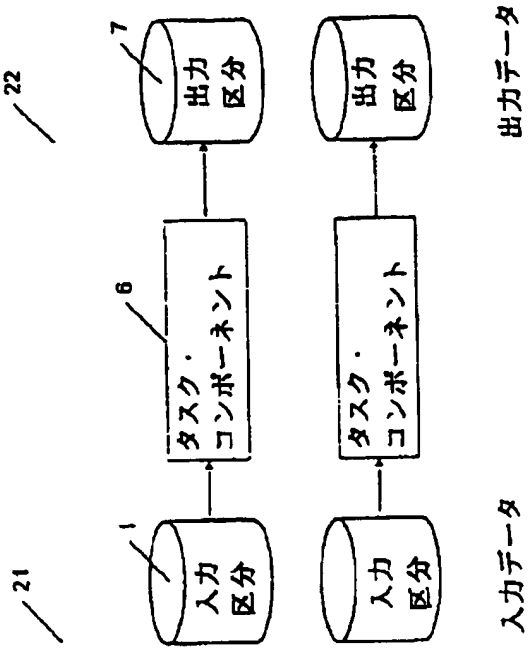



FIG. 2
（従来の技術）

【図3】

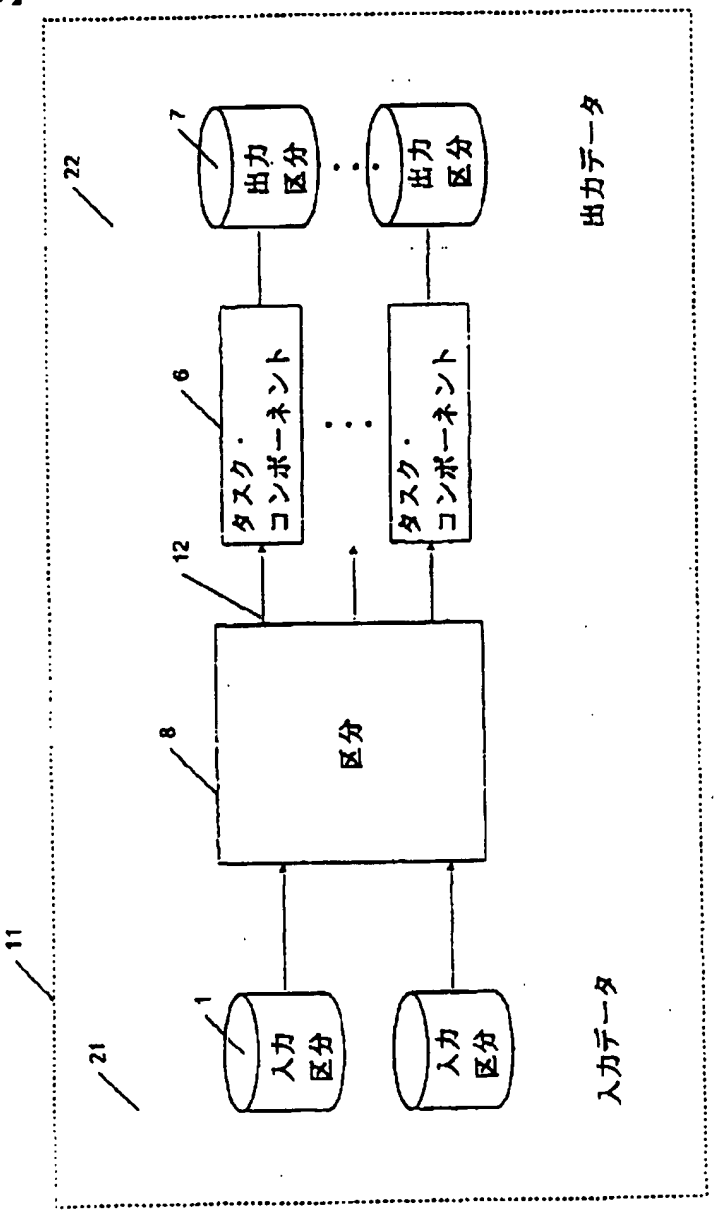



FIG. 3a
(従来の技術)

【図3】

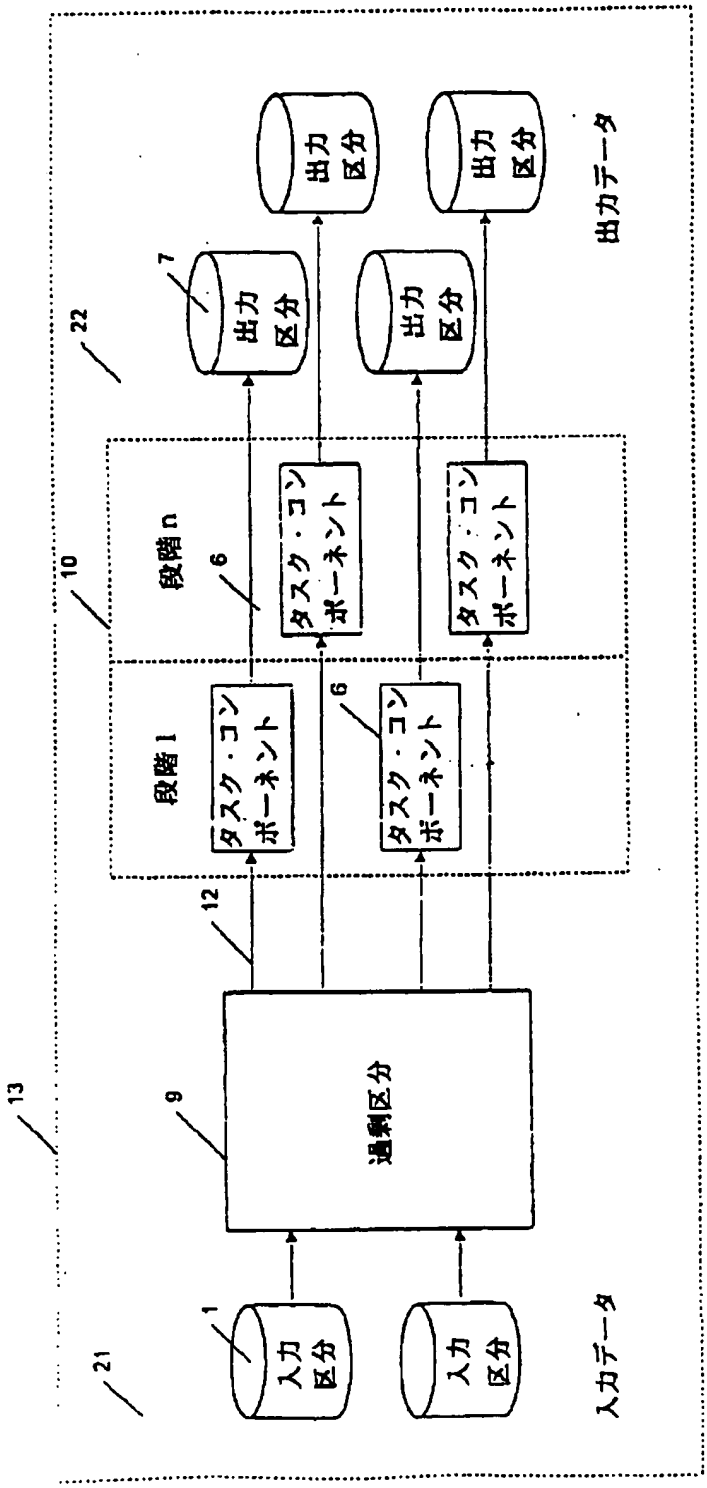

入力区分
入力区分
入力データ
21
1
通信区分
9
13
12
段階1
タスク・コンポーネント
6
タスク・コンポーネント
6
段階n
タスク・コンポーネント
タスク・コンポーネント
10
22
出力区分
出力区分
7
出力区分
出力区分
出力データ

FIG. 3b

【図4】

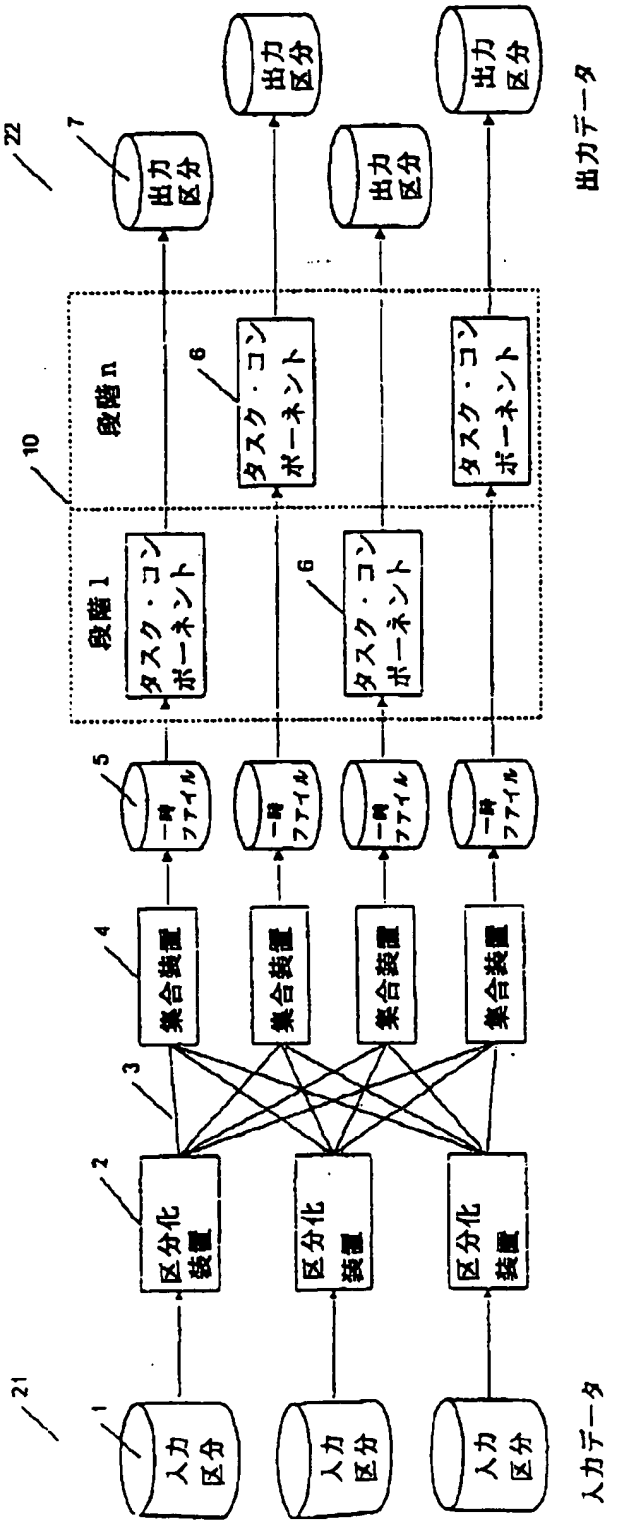

21
22
入力データ
出力データ
入力区分
区分化装置
集合装置
一時ファイル
段階1
段階n
タスク・コンポーネント
出力区分

FIG. 4

【図5】

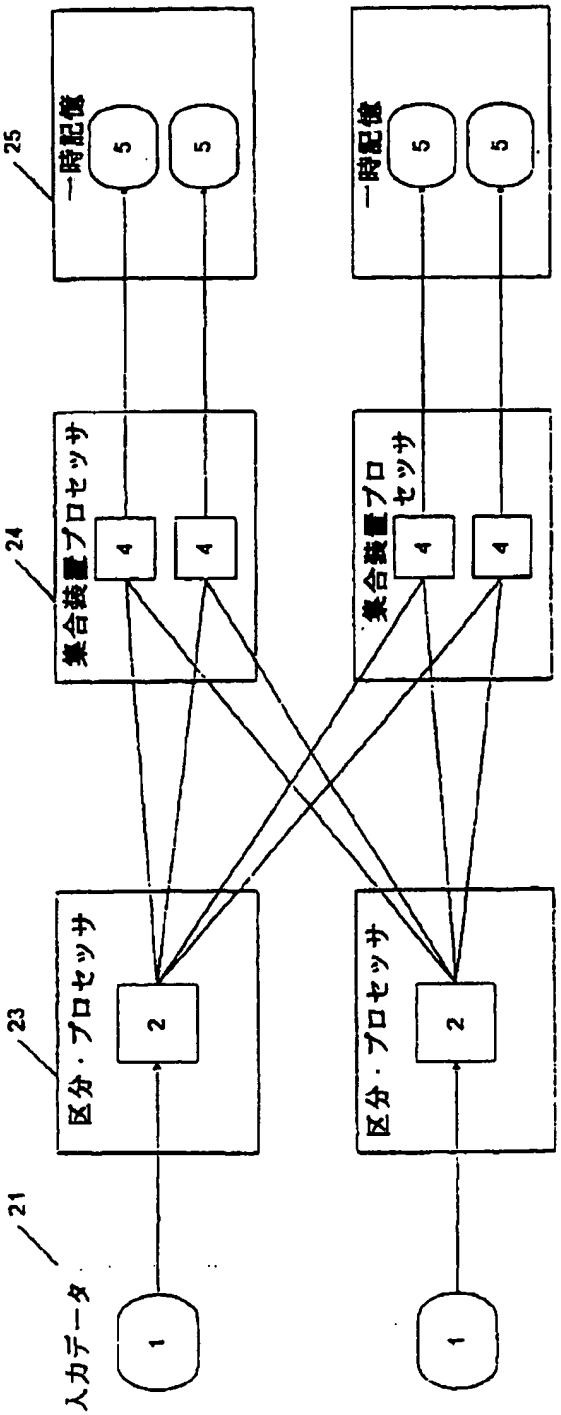



FIG. 5a

【図5】

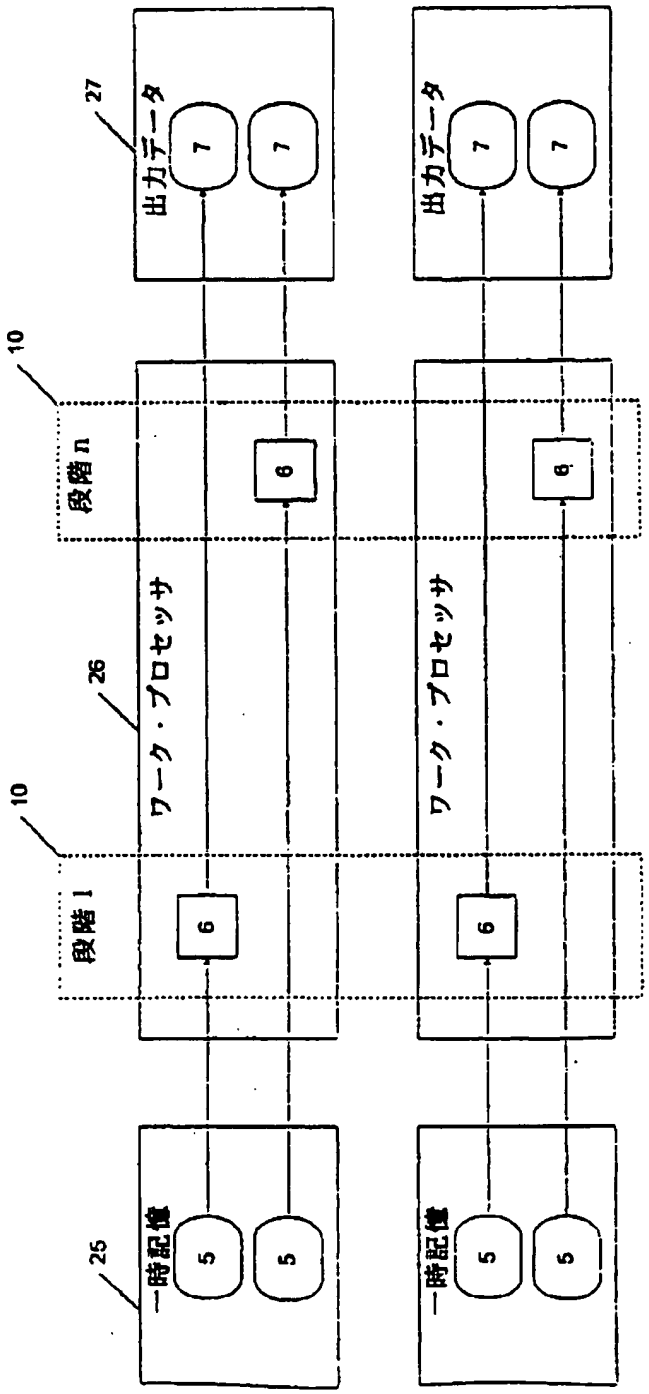



FIG. 5b

【図6】

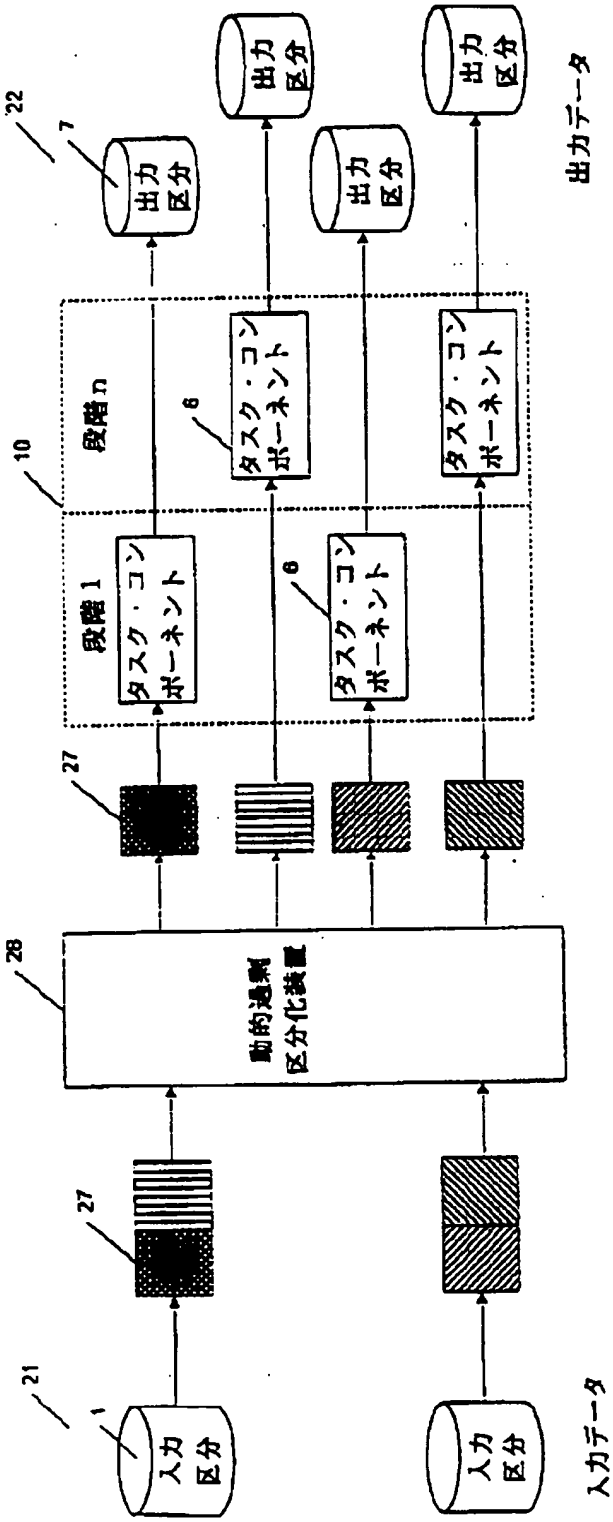

入力データ
入力区分
21
1
27
動的適応区分化装置
28
段階1
タスク・コンポーネント
6
段階n
10
出力区分
22
7
出力データ

FIG. 6

【図7】

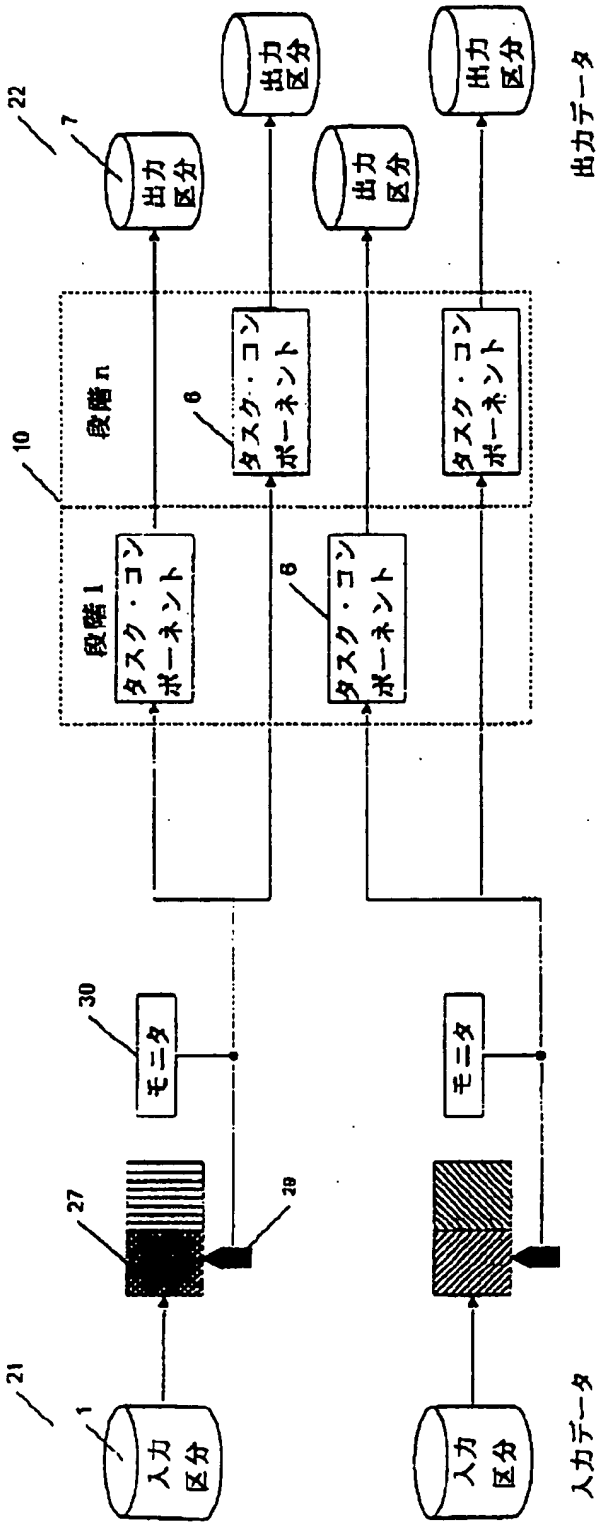



FIG. 7

【図8】

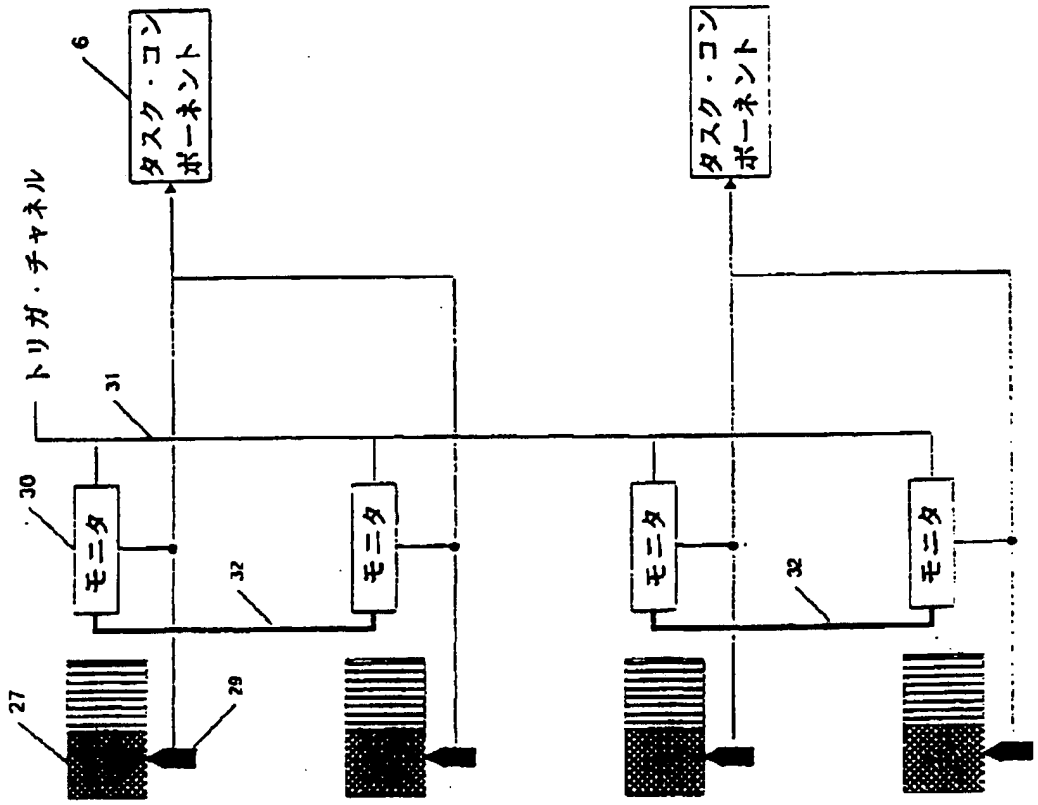

6
タスク・コンポーネント
タスク・コンポーネント
トリガ・チャネル
31
30
モニタ
モニタ
モニタ
モニタ
32
32
27
29

FIG. 8

【手続補正書】

【提出日】１９９８年７月７日

【補正内容】

請求の範囲

１．（ａ）なんらかの整数過剰区分化係数ｎに対して、ｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割することによって、ｑ個のデータ入力を過剰区分化するステップと、

（ｂ）コンポーネント・プログラムの各インスタンスがｎ＊ｐ個のデータ区分の内の１つを読み取り、出力データの対応する区分を作成する、直列のｎ個の実行段階でコンポーネント・プログラムのｐ個のインスタンスを実行するステップと、

（ｃ）ｎ個の実行段階のそれぞれの最後でチェックポイント設定を可能にするステップと、

を含む、並列処理システムで実行可能なコンピュータ・アプリケーションでチェックポイントの頻度を増加するための方法。

２．ｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割するステップが、さらに、

（ａ）データのｎ＊ｐ＊ｑ個のチャネルを出力するように構成される区分化プログラムを、ｑ個のデータ入力に適用するステップと、

（ｂ）ｎ＊ｐ個のデータ区分を生成するために、区分化プログラムによって出力されるデータのｎ＊ｐ＊ｑ個のチャネルに集合プログラムを適用するステップと、

を含む、請求項１記載の方法。。

３．ｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割するステップが、さらに、

（ａ）ｑ個のデータ入力をｎ個のデータ・セグメントに分割するためにｐ個のカーソルを使用するステップと、

（ｂ）各データ・セグメントの最後を示すためにファイル終了信号を回帰的に生成するステップと、

を含む、請求項１記載の方法。。

４．並列処理システム上で実行可能なコンピュータ・アプリケーションでのチェックポイントの頻度を増加するためのコンピュータ・プログラムであって、

（ａ）なんらかの整数過剰区分化係数ｎに対して、ｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割することによって、ｑ個のデータ入力を過剰区分化する機能と、

（ｂ）コンポーネント・プログラムの各インスタンスがｎ＊ｐ個のデータ区分の内の１つを読み取り、出力データの対応する区分を作成する、直列のｎ個の実行段階でコンポーネント・プログラムのｐ個のインスタンスを実行する機能と、

（ｃ）ｎ個の実行段階のそれぞれの最後でチェックポイント設定を可能にする機能と、

を実行するためにコンピュータ・システムによって読み取られ、実行されると、コンピュータ・システムを構成する、コンピュータ・システムによって読取可能な媒体に記録されるコンピュータ・プログラム。

５．ｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割する機能が、さらに、

（ａ）データのｎ＊ｐ＊ｑ個のチャネルを出力するように構成される区分化プログラムを、ｑ個のデータ入力に適用する機能と、

（ｂ）ｎ＊ｐ個のデータ区分を生成するために、区分化プログラムによって出力されるデータのｎ＊ｐ＊ｑ個のチャネルに集合プログラムを適用する機能と、

を含む、請求項４記載のコンピュータ・プログラム。

６．ｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割する機能が、さらに、

（ａ）ｑ個のデータをｎ個のデータ・セグメントに分割するためにｐ個のカーソルを使用する機能と、

（ｂ）各データ・セグメントの最後を示すためにファイル終了信号を回帰的に

生成する機能と、

を含む、請求項４記載のコンピュータ・プログラム。

７．（ａ）なんらかの整数過剰区分化係数ｎに対して、ｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割することによってｑ個のデータ入力を過剰区分化する機能と、

（ｂ）コンポーネント・プログラムの各インスタンスがｎ＊ｐ個のデータ区分の内の１つを読み取って、出力データの対応する区分を作成する、直列のｎ個の実行段階でコンポーネント・プログラムのｐ個のインスタンスを実行する機能と、

（ｃ）ｎ個の実行段階のそれぞれの最後でチェックポイント設定を可能にする機能と、

を実行するために、記録媒体が特定の事前に定義された方法でコンピュータを動作させるように、並列処理システム上で実行可能なコンピュータ・アプリケーションでのチェックポイントの頻度を増加するためにコンピュータ・プログラムで構成されるコンピュータ読取可能な記録媒体。

８．ｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割する機能が、さらに、

（ａ）データのｎ＊ｐ＊ｑ個のチャネルを出力するように構成される区分化プログラムを、ｑ個のデータ入力に適用する機能と、

（ｂ）ｎ＊ｐ個のデータ区分を生成するために、区分化プログラムによって出力されるデータのｎ＊ｐ＊ｑ個のチャネルに集合プログラムを適用する機能と、

を含む、請求項７記載のコンピュータ読取可能な記録媒体。

９．ｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割する機能が、さらに、

（ａ）ｑ個のデータ入力をｎ個のデータ・セグメントに分割するためにｐ個のカーソルを使用する機能と、

（ｂ）各データ・セグメントを示すためにファイル終了信号を回帰的に生成する機能と、

を含む、請求項７記載のコンピュータ読取可能な記録媒体。

１０．コンピュータ読取可能な媒体に常駐し、並列処理システム上で実行可能

なコンピュータ・アプリケーションでのチェックポイントの頻度を増加するためのコンピュータ・プログラムであって、コンピュータに、

（ａ）なんらかの整数過剰区分化係数ｎに対して、ｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割することによって、ｑ個のデータ入力を過剰区分化させ、

（ｂ）コンポーネント・プログラムの各インスタンスがｎ＊ｐ個のデータ区分の内の１つを読み取り、出力データの対応する区分を作成する、直列のｎ個の実行段階でコンポーネント・プログラムのｐ個のインスタンスを実行させ、

（ｃ）ｎ個の実行段階のそれぞれの最後でチェックポイント設定を可能にさせる、

命令を備えるコンピュータ・プログラム。

１１．コンピュータにｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割させる命令が、さらに、コンピュータに

（ａ）データのｎ＊ｐ＊ｑ個のチャネルを出力するように構成される区分化プログラムを、ｑ個のデータ入力に適用させ、

（ｂ）ｎ＊ｐ個のデータ区分を生成するために、区分化プログラムによって出力されるデータのｎ＊ｐ＊ｑ個のチャネルに集合プログラムを適用させる、

命令を含む請求項１０記載のコンピュータ・プログラム。

１２．コンピュータにｑ個のデータ入力をｎ＊ｐ個のデータ区分に機能的に分割させる命令が、さらに、コンピュータに

（ａ）ｑ個のデータをｎ個のデータ・セグメントに分割するためにｐ個のカーソルを使用させ、

（ｂ）各データ・セグメントの最後を示すためにファイル終了信号を回帰的に

生成させる、

命令を含む請求項１０記載のコンピュータ・プログラム。

【国際調査報告】

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US96/19706

A. CLASSIFICATION OF SUBJECT MATTER
IPC(6) :G06F 11/00, 11/08, 11/267
US Cl. :395/118, 181, 673, 674
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED
Minimum documentation searched (classification system followed by classification symbols)
U.S. . 395/118, 181, 673, 674

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
IEEE Data base

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
APS, IEEE Database

C. DOCUMENTS CONSIDERED TO BE RELEVANT

| Category* | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| Y | PEERCY et al, 1995 International Symposium, 27-30 June 1995, "Software Schemes of Reconfiguration and Recovery in Distributed Memory Multicomputers Using the Actor Model" | 1-2, 4-5 and 7-8 |
| Y | US, 5,363,503 A (GLEESON) 08 November 1994, col.2, lns. 1-3, col 7-8, col. 4, lns. 41-52 | 1-9 |

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier document published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search
06 MAY 1997

Date of mailing of the international search report
05 JUN 1997

Name and mailing address of the ISA/US
Commissioner of Patents and Trademarks
Box PCT
Washington, D.C. 20231
Facsimile No. (703) 305-3230

Authorized officer
Kevin Kriscs
Telephone No. (703) 305-3804

Form PCT/ISA/210 (second sheet)(July 1992)*

フロントページの続き

(81)指定国 EP(AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OA(BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG), AP(KE, LS, MW, SD, SZ, UG), UA(AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, HU, IL, IS, JP, KE, KG, KP, KR, KZ, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, TJ, TM, TR, TT, UA, UG, UZ, VN

(72)発明者 レッサー, クリフ
アメリカ合衆国 02139 マサチューセッツ州, ケンブリッジ, リー ストリート 15, アパートメント ナンバー1
(72)発明者 ローディ, ロバート
アメリカ合衆国 01778 マサチューセッツ州, ウェイランド, パウ ロード 33

【要約の続き】

を読み取り、出力データの１つの区分を作成する。

（３）ｎ個の実行段階のそれぞれの最後で、システムは静止しており、チェックポイント設定をすることができる。第１実施例は、過剰区分化された中間ファイルを作り出すために、既知の区分化装置プログラム、通信チャネル、および集合装置プログラムを使用して入力データを明示的に過剰区分化する。第２実施例は、オリジナル入力データの連続サブセットを連続的に読み取るためにコンポーネント・プログラムを手配することによって入力データを動的に過剰区分化する。